

In the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

1 1. (Currently Amended) A method for maintaining coherency of
2 software breakpoints in shared memory when debugging a multiple
3 processor system, the method comprising the steps of:

4 creating a software memory map of the memory usage of a
5 plurality of processors in the system to be debugged;

6 activating a first debug session associated with a first
7 processor of a plurality of processors and at least a second debug
8 session associated with a second processor of the plurality of
9 processors;

10 setting a first software breakpoint in a shared memory
11 location in the first debug session such that all debug sessions
12 are notified of the setting of the breakpoint, the step of setting
13 comprising

14 searching the software memory map to find a first
15 plurality of processors having read access to the shared memory
16 location;

17 updating a software representation maintained for
18 software breakpoints for each of the first plurality of processors;
19 and

20 writing the software breakpoint instruction in the shared
21 memory location; and

22 clearing the first software breakpoint in the shared memory
23 location in the second debug session such that all debug sessions
24 are notified of the clearing of the breakpoint.

2 and 3. (Canceled)

1 4. (Currently Amended) The method of Claim 3 1 wherein the
2 step of writing comprises a method for selecting a processor to
3 execute the write, the method comprising the steps of:
4 if the first processor associated with the first debug session
5 requesting the setting of the software breakpoint has write access
6 to the shared memory location
7 then
8 selecting the first processor to perform the write
9 request;
10 else performing the following steps a-b:
11 a. searching the software memory map for a second
12 processor with write access to the shared memory location;
13 b. selecting the second processor to perform the
14 write request; and
15 passing the software breakpoint instruction to the selected
16 processor to be written into the shared memory location.

1 5. (Original) The method of Claim 4 wherein the step of
2 passing the software breakpoint instruction comprises the steps of:
3 searching the software memory map for a second plurality of
4 processors that have read access to the shared memory location;
5 broadcasting the write request to the second plurality of
6 processors; and
7 performing cache coherency updates in response to the write
8 request in each of the second plurality of processors as required.

1 6. (Original) The method of Claim 5 wherein the step of
2 broadcasting the write request comprises indicating that the write
3 request is intended for maintaining cache coherency as opposed to a
4 normal write request.

1 7. (Original) The method of Claim 6 wherein the step of
2 performing comprises using cache coherency capabilities, if any, of
3 a processor in response to the write request intended for
4 maintaining cache coherency.

1 8. (Currently Amended) ~~The A method of Claim 2 wherein the~~
2 ~~step of clearing comprises~~ for maintaining coherency of software
3 breakpoints in shared memory when debugging a multiple processor
4 system, the method comprising the steps of:

5 creating a software memory map of the memory usage of a
6 plurality of processors in the system to be debugged;

7 activating a first debug session associated with a first
8 processor of a plurality of processors and at least a second debug
9 session associated with a second processor of the plurality of
10 processors;

11 setting a first software breakpoint in a shared memory
12 location in the first debug session such that all debug sessions
13 are notified of the setting of the breakpoint; and

14 clearing the first software breakpoint in the shared memory
15 location in the second debug session such that all debug sessions
16 are notified of the clearing of the breakpoint, the step of
17 clearing comprising

18 writing the original instruction stored in a software
19 representation maintained for software breakpoints into the shared
20 memory location;

21 searching the software memory map to find a fourth
22 plurality of processors having read access to the shared memory
23 location; and

24 updating a software representation maintained for
25 software breakpoints for each of the fourth plurality of processors
26 to remove the software breakpoint for the shared memory location.

1 9. (Original) The method of Claim 8 wherein the step of
2 writing comprises a method for selecting a processor to execute the
3 write, the method comprising the steps of:
4 if the second processor associated with the second debug
5 session requesting the clearing of the software breakpoint has
6 write access to the shared memory location
7 then
8 selecting the second processor to perform the write
9 request;
10 else performing the following steps a-b:
11 a. searching the software representation of the memory map
12 for a third processor with write access to the shared memory
13 location;
14 b. selecting the third processor to perform the write
15 request; and
16 passing the original instruction to the selected processor to
17 be written into the shared memory location.

1 10. (Original) The method of Claim 9 wherein the step of
2 passing the original instruction comprises the steps of:
3 searching the software memory map for a fifth plurality of
4 processors that have read access to the shared memory location;
5 broadcasting the write request to the fifth plurality of
6 processors; and
7 performing cache coherency updates in response to the write
8 request in each of the fifth plurality of processors.

1 11. (Original) The method of Claim 10 wherein the step of
2 broadcasting the write request comprises indicating that the write
3 request is intended for maintaining cache coherency as opposed to a
4 normal write request.

1 12. (Original) The method of Claim 11 wherein the step of
2 performing comprises using cache coherency capabilities, if any, of
3 a processor in response to the write request intended for
4 maintaining cache coherency.

13 to 16. (Canceled)

1 17. (Original) A software development system, comprising:
2 a memory storage system holding a software development tool
3 program;
4 a host computer connected to the memory storage system, the
5 host computer operable to execute the software development tool
6 program;
7 a test port for connecting to a target hardware system, the
8 hardware system being comprised of multiple processors with common
9 shared memory and operable to execute an application program; and
10 wherein the software development tool is operable to support
11 debugging of the application program executing on the target
12 hardware system using a method for maintaining coherency of
13 software breakpoints in shared memory when debugging a multiple
14 processor system, the method comprising the steps of:
15 creating a software memory map of the memory usage of a
16 plurality of processors in the system to be debugged;
17 activating a first debug session associated with a first
18 processor of a plurality of processors and at least a second debug
19 session associated with a second processor of the plurality of
20 processors;
21 setting a first software breakpoint in a shared memory
22 location in the first debug session such that all debug sessions
23 are notified of the setting of the breakpoint, the step of setting
24 comprising:

25 searching the software memory map to find a first
26 plurality of processors having read access to the shared memory
27 location;
28 updating a software representation maintained for
29 software breakpoints for each of the first plurality of processors;
30 and
31 writing the software breakpoint instruction in the
32 shared memory location; and
33 clearing the first software breakpoint in the shared
34 memory location in the second debug session such that all debug
35 sessions are notified of the clearing of the breakpoint.

18 and 19. (Canceled)

1 20. (Currently Amended) The software development system of
2 Claim ~~19~~ 18 wherein the step of writing comprises a method for
3 selecting a processor to execute the write, the method comprising
4 the steps of:
5 if the first processor associated with the first debug session
6 requesting the setting of the software breakpoint has write access
7 to the shared memory location
8 then
9 selecting the first processor to perform the write
10 request;
11 else performing the following steps a-b:
12 a. searching the software memory map for a second
13 processor with write access to the shared memory location;
14 b. selecting the second processor to perform the
15 write request; and
16 passing the software breakpoint instruction to the selected
17 processor to be written into the shared memory location.

1 21. (Currently Amended) A digital system, comprising:
2 multiple processors with common shared memory for executing an
3 application program; and

4 wherein the application program was developed with a software
5 development system using a method for maintaining coherency of
6 software breakpoints in shared memory when debugging a multiple
7 processor system, the method comprising the steps of:

8 creating a software memory map of the memory usage of a
9 plurality of processors in the system to be debugged;

10 activating a first debug session associated with a first
11 processor of a plurality of processors and at least a second debug
12 session associated with a second processor of the plurality of
13 processors;

14 setting a first software breakpoint in a shared memory
15 location in the first debug session such that all debug sessions
16 are notified of the setting of the breakpoint, the step of setting
17 comprising:

18 searching the software memory map to find a first
19 plurality of processors having read access to the shared memory
20 location;

21 updating a software representation maintained for
22 software breakpoints for each of the first plurality of processors;
23 and

24 writing the software breakpoint instruction in the
25 shared memory location; and

26 clearing the first software breakpoint in the shared
27 memory location in the second debug session such that all debug
28 sessions are notified of the clearing of the breakpoint.

22 and 23. (Canceled)

1 24. (Currently Amended) The digital system of Claim 23 21
2 wherein the step of writing comprises a method for selecting a
3 processor to execute the write, the method comprising the steps of:
4 if the first processor associated with the first debug session
5 requesting the setting of the software breakpoint has write access
6 to the shared memory location
7 then
8 selecting the first processor to perform the write
9 request;
10 else performing the following steps a-b:
11 a. searching the software representation of the
12 memory map for a second processor with write access to the shared
13 memory location;
14 b. selecting the second processor to perform the
15 write request; and
16 passing the software breakpoint instruction to the selected
17 processor to be written into the shared memory location.

1 25. (New) A software development system, comprising:
2 a memory storage system holding a software development tool
3 program;
4 a host computer connected to the memory storage system, the
5 host computer operable to execute the software development tool
6 program;
7 a test port for connecting to a target hardware system, the
8 hardware system being comprised of multiple processors with common
9 shared memory and operable to execute an application program; and
10 wherein the software development tool is operable to support
11 debugging of the application program executing on the target
12 hardware system using a method for maintaining coherency of
13 software breakpoints in shared memory when debugging a multiple
14 processor system, the method comprising the steps of:

15 creating a software memory map of the memory usage of a
16 plurality of processors in the system to be debugged;
17 activating a first debug session associated with a first
18 processor of a plurality of processors and at least a second debug
19 session associated with a second processor of the plurality of
20 processors;
21 setting a first software breakpoint in a shared memory
22 location in the first debug session such that all debug sessions
23 are notified of the setting of the breakpoint; and
24 clearing the first software breakpoint in the shared
25 memory location in the second debug session such that all debug
26 sessions are notified of the clearing of the breakpoint, the step
27 of clearing comprising:
28 writing the original instruction stored in a
29 software representation maintained for software breakpoints into
30 the shared memory location;
31 searching the software memory map to find a fourth
32 plurality of processors having read access to the shared memory
33 location; and
34 updating a software representation maintained for
35 software breakpoints for each of the fourth plurality of processors
36 to remove the software breakpoint for the shared memory location.

1 26. (New) The software development system of Claim 25 wherein
2 the step of writing comprises a method for selecting a processor to
3 execute the write, the method comprising the steps of:
4 if the first processor associated with the first debug session
5 requesting the setting of the software breakpoint has write access
6 to the shared memory location
7 then
8 selecting the first processor to perform the write
9 request;

10 else performing the following steps a-b:
11 a. searching the software memory map for a second
12 processor with write access to the shared memory location;
13 b. selecting the second processor to perform the
14 write request; and
15 passing the software breakpoint instruction to the selected
16 processor to be written into the shared memory location.

1 27. (New) A digital system, comprising:
2 multiple processors with common shared memory for executing an
3 application program; and
4 wherein the application program was developed with a software
5 development system using a method for maintaining coherency of
6 software breakpoints in shared memory when debugging a multiple
7 processor system, the method comprising the steps of:
8 creating a software memory map of the memory usage of a
9 plurality of processors in the system to be debugged;
10 activating a first debug session associated with a first
11 processor of a plurality of processors and at least a second debug
12 session associated with a second processor of the plurality of
13 processors;
14 setting a first software breakpoint in a shared memory
15 location in the first debug session such that all debug sessions
16 are notified of the setting of the breakpoint; and
17 clearing the first software breakpoint in the shared
18 memory location in the second debug session such that all debug
19 sessions are notified of the clearing of the breakpoint, the step
20 of clearing comprising:
21 writing the original instruction stored in a
22 software representation maintained for software breakpoints into
23 the shared memory location;

24 searching the software memory map to find a fourth
25 plurality of processors having read access to the shared memory
26 location; and
27 updating a software representation maintained for
28 software breakpoints for each of the fourth plurality of processors
29 to remove the software breakpoint for the shared memory location.

1 28. (New) The digital system of Claim 27 wherein the step of
2 writing comprises a method for selecting a processor to execute the
3 write, the method comprising the steps of:

4 if the first processor associated with the first debug session
5 requesting the setting of the software breakpoint has write access
6 to the shared memory location

7 then

8 selecting the first processor to perform the write
9 request;

10 else performing the following steps a-b:

11 a. searching the software representation of the
12 memory map for a second processor with write access to the shared
13 memory location;

14 b. selecting the second processor to perform the
15 write request; and

16 passing the software breakpoint instruction to the selected
17 processor to be written into the shared memory location.